INT. J. GEOGRAPHICAL INFORMATION SCIENCE, 2001, VOL. 15, NO. 8, 689–705

Research Article

Topology revisited: representing spatial relations

DAVID M. THEOBALD

Natural Resource Ecology Laboratory, Colorado State University, Fort Collins, CO 80523-1499, USA

(*Received 19 April 1999; accepted 6 April 2001*)

Abstract. Topology is a central, defining feature of geographical information systems (GIS). The advantages of topological data structures are that data storage for polygons is reduced because boundaries between adjacent polygons are not stored twice, explicit adjacency relations are maintained, and data entry and map production is improved by providing a rigorous, automated method to handle artifacts of digitizing. However, what explains the resurgence of non-topological data structures and why do contemporary desktop GIS packages support them? The historical development of geographical data structures is examined to provide a context for identifying the advantages and disadvantages of topological and non-topological data structures. Although explicit storage of adjacent features increases performance of adjacency analyses, it is not required to conduct these operations. Non-topological data structures can represent features that conform to planar graph theory (i.e. non-overlapping, space-filling polygons). A data structure that can represent proximal and directional spatial relations, in addition to topological relationships is described. This extension allows a broader set of functional relationships and connections between geographical features to be explicitly represented.

1. Introduction

A core feature of geographical information systems (GIS) is the ability to create and manipulate topological data structures for vector-based data. Topology is typically defined as spatial relationships between adjacent or neighbouring features (DeMers 1997) or '...properties which define relative relationships between spatial elements... including adjacency, connectivity, and containment' (McDonnell and Kemp 1995; p. 88). Therefore, a topological data structure is typically defined as a data structure in which the inherent spatial connectivity and adjacency relationships of features are explicitly stored.

Nearly all GIS textbooks list the advantages of topological data structures as: data storage for polygons is reduced because boundaries between adjacent polygons are not stored twice, explicit neighbourhood relations are maintained, and data entry and map production is improved by providing a rigorous, automated method to handle island and self intersecting polygons, overshoots and undershoots, and gaps (Burrough 1986, Bonham-Carter 1994, Burrough and McDonnell 1998, Chrisman 1997, DeMers 1997). Much of the early GIS research developed and optimized

International Journal of Geographical Information Science
ISSN 1365-8816 print/ISSN 1362-3087 online © 2001 Taylor & Francis Ltd
http://www.tandf.co.uk/journals
DOI: 10.1080/13658810110074519

topological algorithms and data structures (Peucker and Chrisman 1975, Goodchild 1977, Reed 1999). Indeed, topological data structures have been called one of GIScience's '...intellectual breakthroughs of lasting significance' (Goodchild 1992a, p. 37). The data structure used to represent spatial features is important in a GIS because it determines the range of functions and analyses that are easy to provide (Goodchild 1987, Maguire and Dangermond 1991, Raper and Maguire 1992).

However, even though topology has been a central and important concept in GIS since the early-to-mid-1970s (Dueker 1972), why has there been widespread adoption and use of non-topological data structures in contemporary desktop GIS packages? What has led to the claim that a non-topological data format is the *de facto* GIS data transfer standard (Strand 1998)? And why is there so much confusion about the use of topological analyses in non-topological data structures and GIS in general (Reed 1999)?

Here the uses of topology in GIS and the historical development of topological data structures are revisited in an attempt to better understand the advantages and disadvantages of topological and non-topological data structures. In particular, the assumed advantages of pre-computing topological relations are critically examined. Also, an additional line of inquiry is pursued: are there types of spatial relations between features that are not adequately represented in topological data structures. Concepts from graph theory are examined to extend the standard notion of adjacency, and to offer a general data structure to represent spatial relations. The goal in this paper is not to deny the central role of topology in GIS, but to examine critically why and when it is needed, what it means to have topology, what the trade-offs are between topological and non-topological data structures, and to advance the representation of spatial relations. Note that the terminology used below to describe elements of geographical data structures is based on the Spatial Data Transfer Standard (FIPSPUB 173-1, 1994).

2. Cartographic data structures

There are three types of non-topological, or 'unstructured', data structures (Maguire and Dangermond 1991). Spatial relationships between adjacent features are not explicitly stored in all three non-topological data structures. The so-called 'spaghetti' data structure (Chrisman 1977, Peuquet 1984) represents geographical features by a series of points and lines with no systematic correspondence between the points and lines and the geographical features they represent. That is, the boundaries of individual features (lines and polygons) do not necessarily correspond to the chains that represent them (Cromley 1992). The 'primitive instancing' approach uses graphic symbols (e.g. icons) located at x, y locations that represent features such as buildings, roads, traffic lights, etc. as the basic elements stored in the database. The third approach, which is called a cartographic data structure (CDS) here, but is also known as an 'entity-by-entity' data structure, represents geographical features using geometrical objects (points, lines, polygons). Because the first two non-topological data structures do not fully represent the geometry of geographical features, they are inadequate and are excluded from consideration during the subsequent discussion. In a CDS, a point feature is represented by a vertex (a pair of x, y coordinates). A linear feature is represented by a string, which is an ordered sequence of connected, non-branching line segments, where each segment is connected by a straight line connecting two points or vertices. A geometric ring (G-ring) is a

sequence of non-intersecting strings that close. An areal (polygon) feature is represented by an outer ring and 0 or more, non-intersecting, inner G-rings.

An early CDS of note was SYMAP, which encoded features on an entity-by-entity basis (Peucker and Chrisman 1975, Carter 1984). The CALFORM format eliminated the problem of duplicate vertices for shared boundaries by creating a point dictionary, but did not store adjacent neighbors explicitly (Peucker and Chrisman 1975). Many CDS could not properly handle complex, multi-ring ('donut') polygons (Raper and Maguire 1992, Burrough and McDonnell 1998). However, complex polygons can be handled by maintaining a consistent, clockwise ordering of vertices, so that the area to the right (as one 'walks' along the boundary) is inside the polygon and the left is outside (figure 1, table 1). For example, complex polygons are represented using a number of G-rings in a shapefile, which is a commonly-used GIS data format developed by Environmental Systems Research Institute (ESRI 1998).

3. Topological data structures

Although topological relationships among geographical features are implicit in all spatial data, topological data structures (TDS) (figure 2, table 2) explicitly store relationships between adjacent features (DeMers 1997, Burrough and McDonnell

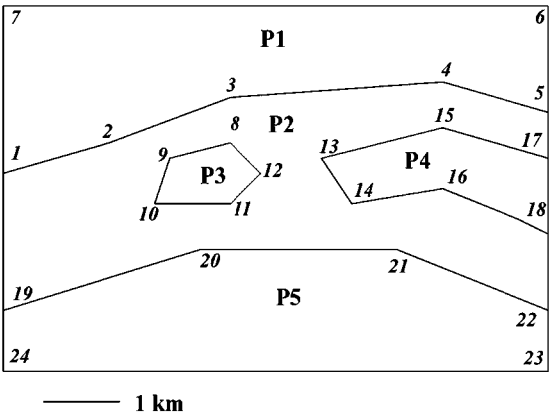

Figure 1. Cartographic data structure. Although this is a simple, non-topological data structure, complex polygons can be represented because 'rings' are stored with their vertices in clockwise order. This allows polygons to track what is 'inside' (to the right) and what is 'outside' (to the left) a polygon.

Table 1. Cartographic data structure of figure 1.

Polygon geometry table		
Polygon	Ring	Vertices
P1	1	1,7,6,5,4,3,2
P2	1	1,2,3,4,5,17,15,13,14,16,18,22,21,20,19
P2	2	8,9,10,11,12
P3	1	8,12,11,10,9
P4	1	17,18,16,14,13,15
P5	1	19,20,21,22,23,24

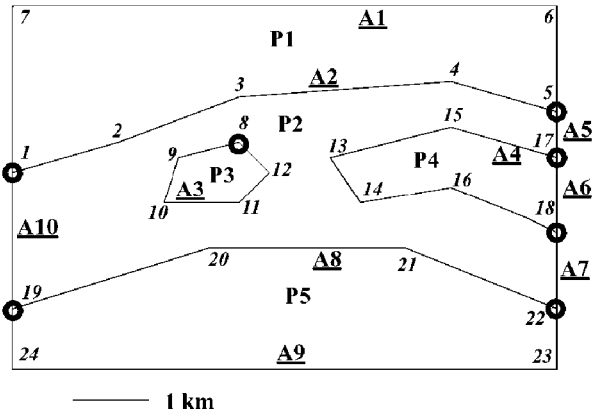

Figure 2. Topological data structure. Vertices are numbered from *1* through *24*, and nodes are denoted by circles. Chains are labelled *A1* through *A10*, while polygons are labelled *P1* through *P4*.

1998). Combinatorial topology is used to define relationships in spatial data, where nodes (0-cells), edges (1-cells), and faces (2-cells) on a plane form a linear graph (Alexandroff 1961, Cooke and Maxfield 1967, Corbett 1975, 1979). Nodes are features of 0 dimension (a point) where one or more chains connect to form a topological junction. A chain is a directed, non-branching sequence of non-intersecting line segments bounded by nodes. Chains reference the left and right polygons and start and end nodes. Topological rings (GT-rings) are created from a sequence of non-intersecting chains that close (i.e. the first and last vertices are the same). A topological polygon is defined by GT-rings created from its bounding chains. The universe polygon lies outside the perimeter of the area covered by other GT-polygons and is needed to complete the adjacency relationships of the perimeter polygons. By requiring a node at the intersection of two linear features, planar topology is enforced and results in a single set of non-overlapping features (lines and polygons).

There are two main types of TDS: directional and complex (Maguire and Dangermond 1991, Raper and Maguire 1992). An example of the directional approach is the well-known Dual Independent Map Encoding (DIME) system, developed to process the 1970 US Census. DIME is widely considered to be the first explicit TDS and relied on the application of topology to reduce the 'optical clutter' of common boundaries represented by duplicate lines and to detect data entry errors in the database (Cooke and Maxfield 1967, Peucker and Chrisman 1975, Clarke 1990, Chrisman 1997). A further innovation in TDS was contributed by the POLYVRT data structure. POLYVRT is an example of a 'complex' TDS and was designed to handle more complicated features by replacing a single-line segment (that DIME used) with a chain to represent an edge. It also explicitly stored adjacent polygons (Peucker and Chrisman 1975). In addition, the relationship between neighbouring features was thought to be a fundamental characteristic of geographical data, and it was argued that typical GIS analysis would require explicit topology to ensure adequate performance for geographical analyses. Also, by pre-processing topology rather than computing it on-the-fly, the geographical data could be separated from the application program (Peucker and Chrisman 1975). A wide-variety of 'complex' TDS have been developed and implemented in the three decades since the

Table 2. Topological data structure of figure 1.

Arc geometry table

Arc	Vertices
A1	1,7,6,5
A2	1,2,3,4,5
A3	8,9,10,11,12
A4	17,15,13,14,16,18
A5	5,17
A6	17,18
A7	18,22
A8	19,20,21,22
A9	22,23,24,19
A10	1,19

Arc topology table

Arc	Left polygon	Right polygon
A1	—	P1
A2	P1	P2
A3	P2	P3
A4	P2	P4
A5	—	P2
A6	—	P4
A7	—	P2
A8	P2	P5
A9	—	P5
A10	P2	—

Polygon topology table

Polygon	Adjacent polygons
P1	P2
P2	P1,P3,P4,P5
P3	P2
P4	P2
P5	P2

seminal work on topological data structures, including CANSIS (Tomlinson 1967, Tomlinson 1998), TIGER (Marx 1986, Cooke 1998), and ARC/INFO coverages (Morehouse 1992).

4. Advantages and disadvantages

In general terms, the same characteristics and advantages of explicit storage of topology in GIS have remained constant. That is, duplicate polygon boundaries are not repeated; errors introduced during map digitizing and data entry can be automatically checked; and analyses that require adjacency, containment, and connectivity can utilize explicit topology to provide adequate performance (Bonham-Carter 1996, Chou 1997, Burrough and McDonnell 1998, DeMers 1997). An additional result of enforcing planar topology (Goodchild 1992b) is that although properties of geometric

features are not invariant under a strict topological definition, in practice the sum of the area of a group of individual (space-filling) polygons equals the total area of the region covered by those polygons (Goodchild and Kemp 1990).

A disadvantage of TDS is that the topological tables must be created in the first place (whether they are used or not), requiring not only computational time, but often multiple edit sessions to remove under- and over-shoots, and sliver polygons (Aronoff 1989). Graphic display of geographical data stored in TDS is slower because the chains (and vertices that make up chains) that constitute a geographical feature are not typically stored sequentially, and therefore must be extracted from different data structures or files (Bonham-Carter 1996). Three-dimensional geographical features, such as overpasses and tunnels, and complex features, such as one-way streets, self-intersecting transportation routes (e.g. bus routes), and parcels represented by disjoint polygons, cannot be represented in a strict planar graph (Raper and Maguire 1992). These complex, but relatively common, features require an extension of standard line/polygon data modeling (ESRI 1995). It is also interesting to note that, in practice, few commercial GIS packages provide direct access to topological neighbours (Chrisman 1997).

An advantage of CDS over TDS is that they can be drawn and edited faster because all the geometric features are stored sequentially in one file. The file configuration of CDS is simple and lends itself to being 'open' (Aronoff 1989, ESRI 1998), allowing other software applications to use geographical data (Strand 1998).

There are a number of disadvantages of CDS that are typically cited. First, adjacent polygons that share common boundaries duplicate common vertices and therefore file sizes are expected to be larger (Burrough and McDonnell 1998). Data files that represent complex linear and areal geographical features (with many vertices) can be theoretically up to twice as large. However, in practice, file sizes are rarely twice as large as TDS, partly because polygon segments on the border of a map are not duplicated, because TDS require additional files to store the topological information, and because attribute tables often are a large proportion of the overall file size (and are roughly the same size in CDS and TDS). Another limitation of duplicating common polygon boundaries is that generalizing (simplifying) a boundary (e.g. Douglas and Peucker 1973) will likely create slivers between the boundaries.

A second disadvantage of CDS typically identified in the literature is that adjacency, containment, and connectivity analyses are severely limited (Cowen 1988, Maguire and Dangermond 1991). Much of the confusion about the limitations of CDS comes from the assumption that topological relationships must be stored to conduct any operation that requires relationships between adjacent features. Many GIS operations can be performed faster if explicit adjacency relationships are precomputed. However, even though CDS do not store explicit adjacencies, adjacency operations can be carried out by computing spatial intersections of geographical features on-the-fly, with adequate performance for the average GIS user. For example, the geometric intersection of any pair of polygons can be computed and described by the eight-relations model for 2-dimensional features—disjoint, contains, inside, equal, meet, covers, covered-by, and overlap (Egenhofer and Herring 1990). If the relation type is anything but 'disjoint', then the pair would be considered adjacent. A second assumed limitation is that planar topology is required to create the spatial relationships stored in explicit data structures. The 'meet' relations type is only one of the seven relations that would constitute adjacency (the eighth relation, 'disjoint', still would not constitute adjacency).

A third (implied) disadvantage of CDS is that robust map creation and editing cannot be accomplished. Because standard digitizing procedures result in 'cartographic spaghetti', TDS were developed to provide a rigorous, automated method to clean up data entry errors and verify data (Chrisman 1987). The typical digitizing approach is to first digitize all the lines on a map. Then, planar graph theory is used to identify self-intersecting polygons, islands, overshoots and undershoots, and slivers and gaps (Chrisman 1997, Burrough and McDonnell 1998). Individual polygons and line features are then labelled.

However, instead of a sequential process where all features are digitized *en masse* and planar topology is enforced to build points, lines, and polygons, a map can be created through a feature-based approach where the boundary of a single feature is digitized and then labelled. New features can be automatically clipped to existing adjacent features, ensuring that adjacent features share vertices along the common boundary (or point for lines). Possible gaps between features can also be identified on-the-fly. This feature-based approach was unavailable in the past because computer processing and graphic display was too slow (White *et al.* 1987).

5. Spatial relations

Although topology is the best-known spatial relation type, there are two other commonly recognized types of spatial relationships between geographical features: proximal and directional (Jones 1997). Proximal relationships describe the distance between geographical features. Directional relationships describe above-below and cardinal directions between features (e.g. Peuquet and Zhan 1987). Even though distance-based queries are very common in GIS, proximal relationships are not stored in a typical topological data structure (except when proximal is defined as 0, when proximal and adjacent relationships are equivalent). A related concept is the object pair, a virtual object that represents some relationship between two geographic features (Goodchild 1987, 1992b).

There are a number of applications and uses of computing connectivity through proximal relations. First, artifacts, or accidental geographical features that commonly result from editing and planar enforcement of geographical data models, can be overcome. For instance, sliver polygons (i.e. small, thin polygons created at the boundary of polygons) often cause difficulties during analysis if the distance between two features (D) is greater than 0, as occasionally two features are not topological neighbours (and do not meet).

For example, the analytical operation to find neighbouring polygons would be adjusted so that polygons within distance D (defined as the spatial precision of the vertices) would be selected as well as the directly adjacent ('meet' case) polygons. This modification would make operations more robust to data sets that contain features that have small gaps or slivers between them. Moreover, this will resolve the paradox that the more accurately boundaries are defined during digitizing, the greater the number of spurious polygons are created (Goodchild 1977).

Second, defining connectivity solely on the basis of topological adjacency is inadequate for studying many types of natural and social processes that exhibit scale dependency (Wiens *et al.* 1993, Theobald and Hobbs 2001). For instance, vegetation patches are typically assumed to be equivalent to mapped polygons. However, patches may be functionally connected or contiguous even if they are spatially disjunct, but are within some distance related to a given process, such as the movement of an animal across a landscape (Bunn *et al.* 2000). Also, some processes

are connected but are constrained further than simple topologic adjacency. For example, water basins (represented by polygons) are functionally connected only if they are adjacent and water flows from one into another (figure 3). Two polygons that share a common boundary located on a ridge are not functionally connected, and connectivity in this sense is uni-directional, not bi-directional. Third, connectivity defined in terms of proximal relations can be used to examine structure in geographical phenomena represented by points, in addition to lines and polygons. Although partitioning space into proximity regions using Thiessen polygons is often used to define proximal relations (Cromley 1992), proximity here is not restricted to representations of data using planar topology.

Graph theory, of which planar graphs are a special case (Gibbons 1985, Frank 1992), provides a basis to develop a more general approach to describing proximal spatial relationships. First, the set of connected features in graph theory is not strictly limited to topologically adjacent features—potentially all features can be connected. A key feature in graph theory is the adjacency matrix, which is used to represent the connectivity structure of a graph. Here the adjacency matrix is called the connectivity matrix in order to avoid confusion with the more limited case of topological adjacency. A connectivity matrix is an $n \times n$ matrix, where n is the number of geographical features (called nodes in graph theory). A 1 is placed at each cell of the matrix if feature i and j are connected, 0 if they are not. For example, table 3(*a*) represents the topological adjacency connectivity of figure 2. Suppose that polygon 2 is actually a river, and that the remaining polygons (islands and mainland) can be connected via bridges. However, further suppose that it is only practical to build bridges that span up to 1 km. In other words, the land masses are functionally connected if they are less than 1 km apart. This definition incorporates proximal spatial relationships to establish connectivity and would result in the connectivity table reflected in table 3(*b*) (note all land bodies are connected except 1 and 5). It is

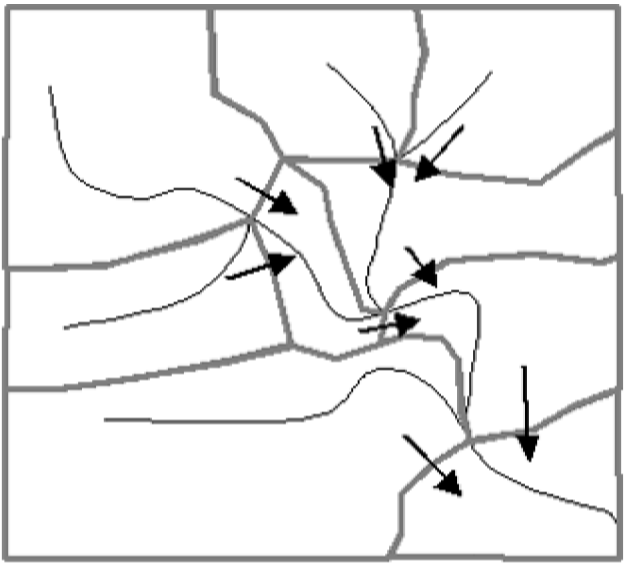

Figure 3. Watershed connectivity. Polygons representing watersheds are shown by thick grey lines, the stream network is shown by thin black lines, and the connected watersheds ('flow') are shown by arrows.

Table 3. Adjacency matrices for figure 1.

	Topological adjacency				
	P1	P2	P3	P4	P5
P1	—	1	0	0	0
P2	1	—	1	1	1
P3	0	1	—	0	0
P4	0	1	0	—	0
P5	0	1	0	0	—
	Bridges spanned connectivity				
	P1	P2	P3	P4	P5
P1	—	0	1	1	0
P2	0	—	0	0	0
P3	1	0	—	1	1
P4	1	0	1	—	1
P5	0	0	1	1	—

clear that by modifying the proximity threshold, the scale dependency of the map can be examined as well.

Based on the adjacency lists of Gibbons (1985), a more reasonable data structure can be developed that does not explicitly store a record for features that are unconnected (0 in the connectivity matrix). The list is broken into two sections and employs an indexed sequential access method (tables 4(*a* and *b*)). The top n records store an index to point to the first record of the connected features, while the remaining records store the IDs of the connected features. The number of records in this data structure required to represent connectivity ranges from $n+1$ (where there is only 1 connection between features) to n^2+n (for a completely connected graph). The number of features equals the value in the third column, first record minus 1, $V_{3,1}-1$. To find the features that are connected to feature j, retrieve the value in the third column: $V_{3,j}=7$. For example, to get the connected features of feature 2, begin at row 7, and step through the rows $(V_{3,2+1}-V_{3,2})$ times. The number of features connected to feature j is $V_{3,j+1}-V_{3,j}$. In practice, only the third column needs to be represented in a file, the first two columns are presented for clarity. Note that each record stores a uni-directional connection. Object pairs, including their attributes, can be represented in this data structure as well (with the addition of a column per attribute).

6. Algorithms to compute connectivity

Analytical operators that require connectivity information will perform faster when a connectivity table exists. Three algorithms to compute connectivity using proximal spatial relationships for geographical phenomena represented by point, line, and polygon feature types are presented. The algorithms are presented in pseudo-code (see the Appendix). The results of these algorithms can then be stored in a connectivity table.

These algorithms and the supporting discussion are all based on computing the distance between two features of the same dimension (i.e. point, lines, areas).

Table 4. Connectivity data structure for figure 1. The first six lines store pointers to the first connected feature. For example, to find the connected features of polygon 1, the first feature is defined by the value in the 'connected feature' field on record 1 (7 in this example). The connected feature field in record 7 is 2, showing that polygon 2 is connected to polygon 1. Note that the last record in the header (#6) defines the end of file with a feature number = −9.

Data structure storing topological adjacency		
Record number	Feature number	Connected feature
1	1	7
2	2	8
3	3	12
4	4	13
5	5	14
6	−9	15
7	1	2
8	2	1
9	2	3
10	2	4
11	2	5
12	3	2
13	4	2
14	5	2

Data structure storing bridge-connected		
Record number	Polygon number	Connected feature
1	1	7
2	2	9
3	3	9
4	4	12
5	5	15
6	−9	17
7	1	3
8	1	4
9	3	1
10	3	4
11	3	5
12	4	1
13	4	3
14	4	5
15	5	3
16	5	4

Typically, the distance between lines (and areas) is computed as the centroid-to-centroid distance. Here distance equals the minimum distance between any pair of points, lines or areas is assumed. In addition to computing minimum distance, the average and maximum distance between pairs of the same feature type can be calculated (Okabe and Miller 1996). Again, the standard notion of adjacency as 'the sharing of a common boundary of two regions or polygons' (McDonnell and Kemp 1995, p. 7) is revised here to include all topological relationships for 2-dimensional features defined in the 8-relations model (Egenhofer and Herring 1990)—disjoint,

contains, inside, equal, meet, covers, covered-by, and overlap. When examining exact-adjacency (distance between features is equal to 0), it is important to note that these algorithms will find features that can intersect only at a point (0-meet), in addition to intersecting a line (1-meet) that is reported by standard topological definitions (Egenhofer and Herring 1990).

A simple algorithm to compute connectivity is the so-called 'brute-force' approach (see Appendix). For every pair of features, determine if they are within a defined distance, and then store the adjacent index values. The time required (worst-case) for this algorithm is proportional to the square of the number of features (n), or order $O(n^2)$. Because connectivity is a reflexive spatial relation (Freeman 1975), the brute-force algorithm can be modified to store the reciprocal feature's index as well. The time required for this algorithm is proportional to $O(n(n-1)/2)$. A third algorithm uses the 'divide and conquer' approach to build a spatial index to subdivide the features into smaller groups, for which the reflexive brute-force approach is then applied. Because the number of features in each bin is much smaller than n for data sets with a large n, the overall number of intersection tests between two features is reduced considerably. Rather than subdividing the map extent into a prescribed number of bins, or cells in a uniform grid (Pullar 1990), features can be recursively divided or 'quad-tiled' to reduce the number of comparisons needed by the brute-force algorithm.

The algorithm begins with a list of all features n within the map extent. The extent is subdivided into four equal-area tiles and a new list of features found in each tile is generated. If the number of features found in a tile is greater than a user-defined threshold t, then the tile is recursively subdivided again into four tiles. Otherwise, the list of features is placed in a stack. Once all tiles have less than t features, then the reflexive brute-force approach is used to create the adjacency list. The optimal value for t depends on how regular the map features are decomposed, but, in practice, a reasonable number for t typically ranges from 25 to 50. This algorithm computes in time nearly proportional to $O(n \log(n))$ (Huber 2000).

7. Discussion

If existing approaches are the result of compromises and constraints largely influenced by previous technology (Copeland 1982, Blakemore 1984, Goodchild 1988), then GIS users need to refine their understanding of adjacency and connectivity, and of topological and non-topological data structures. At a fundamental level, revisiting the need for and use of topology in GIS raises a classic dilemma in computer programming—how to balance technological limitations of data storage costs and processing time with presumptions about GIS user requirements. We can only speculate, but the technological advances in processor speed that allow on-the-fly calculation of feature intersections combined with the limited set of functions required by typical GIS users has allowed a resurgence of CDS. Two decades ago when the relative merits of topological and non-topological data structures were wrestled with, it was clear that the balance tilted in favour of explicit storage of topology to provide adequate performance for spatial analysis. However, given that processing power has doubled every 18 months (Moore's law), pre-processing data is no longer a requirement for adequate performance for many GIS users. Moreover, in the past it was argued that GIS users would need topology for complex geographical analyses, but the majority of routine uses of current commercial GIS do not require complex analyses (Martin 1996). Indeed, the emergence of desktop GIS has

further eroded the need for high-level analysis by the 'average' GIS user. Rather, simple cartographic display and query is more common.

In addition, there are a number of advantages of CDS that provide functionality not easily available through TDS. In situations where multiple, but closely related, attributes occur at a single location, CDS are easier to manipulate and analyse because *objects* in the target domain (database representation) correspond more closely to the *entities* in the source domain (real world phenomena) they represent (Raper and Maguire 1992, Worboys 1995). That is, 'It appears that an entity orientation is more natural than thinking of regular partitions of space...' (Laurini and Thompson 1992, p. 254). For example, elk habitat can be represented in a polygon CDS by overlapping polygons (table 5(*a*)), where each polygon represents a different type of elk range (figure 4). Representing the same situation using TDS requires either multiple layers (each range type in a different coverage) or a single layer with multiple attributes (table 5(*b*)). Another example is US Public Land Survey System townships and sections, which can be represented in a CDS as polygons. Townships can be distinguished from sections by using a different outline symbol, whereas townships and sections must be in separate layers to accomplish this in a TDS.

A major challenge in GIS is to represent temporal changes in geographical phenomena (Langran 1992). Generally, TDS layers are 'snapshots' of a state at a particular time, and historical conditions exist in archived layers. For example, county parcel maps (land ownership) show current conditions, and the plethora of changes (ownership changes, lot splits, new subdivisions) are essentially lost. A polygon CDS could be used to track parcel changes through time, conceptually adding a new parcel to the 'stack' for each change. A parcel query would be facilitated by such as structure. This concept can be extended to feature edits and updates, replacing the temporal attribute with a version value.

Many useful, appropriate analyses do not require data with planar topology. For

Table 5. Attribute tables for a non-topological and topological data structures in figure 4.

(*a*) The attribute table for non-topological representation of figure 4. Note that the summation of the three polygons exceeds the total overall habitat.

Polygon	Habitat
1	Winter
2	Severe
3	Concentration

(*b*) The attribute table for the topological representation of figure 4. Note that because each polygon can have any of the three attributes (winter, severe, concentration area), each polygon has multiple attributes.

Polygon	Winter	Severe	Concentration
1	yes	no	no
2	yes	yes	no
3	no	no	yes
4	yes	no	no

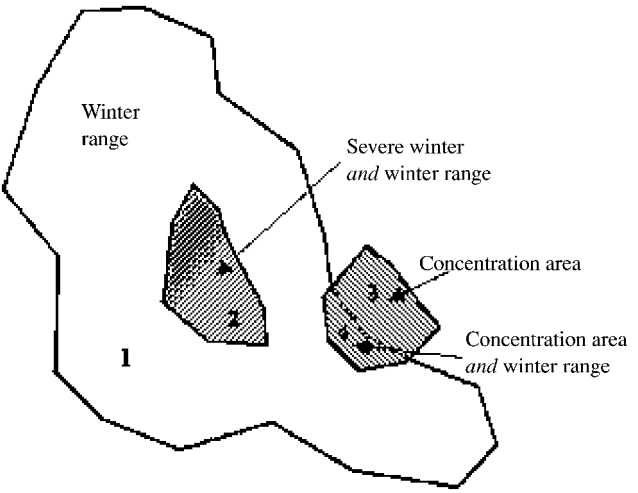

Figure 4. Representing features using topological and non-topological structures.

example, local governments have found it extremely time-consuming and difficult to build parcel layers because parcel boundaries rarely match cleanly with adjacent parcels, and resolving these boundary disputes is a very time consuming process (and often fraught with complicated legalities). Two important and common uses of parcel data do not require planar enforced data—finding the landowner at a given location and identifying all the neighbours within 150 m (500 ft) of that parcel.

8. Conclusion

It is worth repeating that the data structure used to represent spatial features in a GIS is important because it determines the range of functions that are easy to provide (Goodchild 1987, Maguire and Dangermond 1991, Raper and Maguire 1992). Topological data structures remain central to GIS, and, as has been shown, they are optimized for representing data that conform to planar graphs and for conducting analyses that require topological adjacency. Explicit storage of topologically-adjacent features increases performance of adjacency analyses but is *not* required to conduct these operations. Planar topology provides a rigorous method to identify data entry errors during spaghetti-digitizing, allows users to calculate areas assuming space-filling polygons, and identifies adjacent features that are touching (lines) or share an edge (polygons). However, planar enforcement is independent of explicit storage of spatial relationships— features stored in a non-topological data structure can also conform to planar graphs through 'WYSIWYG' feature-based digitizing and editing. A more precise definition of topological data structures is: topological data structures explicitly represent exactly-adjacent features derived from applying planar graph theory to spatial features.

In addition to topological spatial relations, proximal relations can be computed and represented using the connectivity data structure. Indeed, this data structure offers the opportunity to expand the notion of connectivity as exactly-adjacent, to represent a full range of spatial relations, including the object pair. An important opportunity provided by the connectivity data structure is that the relationship between the structure and the functional connectivity of a map can be examined.

Acknowledgments

The author thanks Pete Weisberg, Randy Boone, and Chris Bennett for their helpful comments on earlier drafts, and to the thoughtful comments of the anonymous reviewers that strengthened the arguments and focus of this paper.

References

Alexandroff, P., 1961, *Elementary Concepts of Topology* (New York: Dover Publishing Inc).

Aronoff, S., 1989, *Geographic Information Systems: A Management Perspective* (Ottawa, Ontario: WDL Publications).

Blakemore, M., 1984, Generalization and error in spatial databases. *Cartographica*, **21**, 131–139.

Bonham-Carter, G. F., 1996, *Geographic Information Systems for Geoscientists: Modeling with GIS* (Ottawa, Ontario: Pergamon Press).

Bunn, A. G., Urban, D. L., and Keitt, T. H., 2000, Landscape connectivity: A conservation application of graph theory. *Journal of Environmental Management*, **59**, 265–278.

Burrough, P. A., 1986, *Principles of Geographical Information Systems for Land Resources Assessment* (Oxford: Clarendon Press).

Burrough, P. A., and McDonnell, R. A., 1998, *Principles of Geographical Information Systems* (Oxford: Oxford University Press).

Carter, J. R., 1984, *Computer Mapping: Progress in the '80s* (Washington, DC: Association of American Geographers).

Chou, Y. H., 1997, *Exploring Spatial Analysis in Geographic Information Systems* (Sante Fe, New Mexico: Onword Press).

Chrisman, N., 1975, Topological data structures for geographic representation. In *Proceedings of the International Symposium on Computer-Assisted Cartography: Auto-Carto II* (Bethesda, MD: American Congress on Surveying and Mapping), pp. 346–351.

Chrisman, N., 1977, *Impact of Data Structure on Geographic Information Processing*. Internal Report no. 7404, Laboratory for Computer Graphics and Spatial Analysis, Harvard University.

Chrisman, N., 1987, Efficient digitizing through the combination of appropriate hardware and software for error detection and editing. *International Journal of Geographical Information Systems*, **1**, 265–277.

Chrisman, N., 1997, *Exploring Geographic Information Systems* (New York: John Wiley & Sons).

Clarke, K. C., 1990, *Analytical and Computer Cartography* (Englewood Cliffs: Prentice Hall).

Cooke, D. F., 1998, Topology and TIGER: The Census Bureau's contribution. In *The History of Geographic Information Systems: Perspectives from the Pioneer*, edited by T. W. Foresman (Upper Saddle River, NJ: Prentice Hall), pp. 47–57.

Cooke, D. F., and Maxfield, W. H., 1967, The development of a geographic base file and its uses for mapping. *Urban and Regional Information Systems Association*, **5**, 207–218.

Copeland, G., 1982, What if mass storage were free? *IEEE Computer*, **15**, 27–35.

Corbett, J. F., 1975, Topological principles in cartography. In *Proceedings of the International Symposium on Computer-Assisted Cartography: Auto-Carto II* (Bethesda, MD: American Congress on Surveying and Mapping), pp. 61–65.

Corbett, J. F., 1979, *Topological Principles in Cartography*. Technical paper 48, US Department of Commerce, Bureau of the Census: Washington, D.C.

Cowen, D. J., 1988, GIS versus CAD versus DBMS: what are the differences? *Photogrammetric Engineering & Remote Sensing*, **54**, 1551–1555.

Cromley, R. G., 1992, *Digital Cartography* (Englewood-Cliffs, N.J., Prentice-Hall).

DeMers, M. N., 1997, *Fundamentals of Geographic Information Systems* (New York: John Wiley & Sons).

Douglas, D. H., and Peucker, T. K., 1973, Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. *The Canadian Cartographer*, **10**, 112–122.

Dueker, K. J., 1972, A framework for encoding spatial data. *Geographical Analysis*, **4**, 98–104.

Egenhofer, M. J., and Herring, J. R., 1990, A mathematical framework for the definition of

topological relationships. In *Proceedings of the Fourth International Symposium on Spatial Data Handling*, International Geographical Union, Zurich 1990, pp. 803–813.

ESRI, 1995, *Regions: A powerful polygon modeling tool for representing complex areas* (Redlands, CA: Environmental Systems Research Institute).

ESRI, 1998, *Shapefile technical description* (Redlands, CA: Environmental Systems Research Institute).

Frank, A. U., 1992, Spatial concepts, geometric data models, and geometric data structures. *Computers & Geosciences*, **18**, 409–417.

Freeman, J., 1975, The modeling of spatial relations. *Computer Graphics and Image Processing*, **4**, 156–171.

Gibbons, A., 1985, *Algorithmic Graph Theory* (Cambridge: Cambridge University Press).

Goodchild, M. F., 1977, Statistical aspects of the polygon overlay problem. In *Proceedings of the First International Advanced Study Symposium on Topological Data Structures for Geographic Information Systems* (Harvard: Laboratory for Computer Graphics and Spatial Analysis, Graduate School of Design, Harvard University).

Goodchild, M. F., 1987, A spatial analytical perspective on geographical information systems. *International Journal of Geographical Information Science*, **1**, 327–334.

Goodchild, M. F., 1988, Stepping over the line: Technological constraints and the new cartography. *The American Cartographer*, **15**, 311–319.

Goodchild, M. F. 1992a, Geographical Information Science. *International Journal of Geographical Information Systems*, **6**, 31–45.

Goodchild, M. F., 1992b, Geographical data modeling. *Computers & Geosciences*, **18**, 401–408.

Goodchild, M. F., and Kemp, K. K., editors, 1990, *NCGIA Core Curriculum in GIS*. National Center for Geographic Information and Analysis, University of California, Santa Barbara CA.

Huber, W., 2000. Select overlapping polygons script. http://gis.esri.com/arcscripts/details.cfm?CFGRIDKEY=950305132.

Jones, C. B., 1997, *Geographical Information Systems and Computer Cartography* (Harlow: Longman).

Langran, G., 1992, *Time in Geographic Information Systems* (London: Taylor & Francis).

Laurini, R., and Thompson, D., 1992, *Fundamentals of Spatial Information Systems* (New York: Academic Press).

McDonnell, R., and Kemp, K. K., 1995, *International GIS Dictionary* (New York: GeoInformation International).

Maguire, D., and Dangermond, J., 1991, The functionality of GIS. In *Geographical Information Systems: Principles and Applications*, edited by D. J. Maguire, M. F. Goodchild and D. W. Rhind (Harlow: Longman Scientific & Technical), pp. 319–335.

Martin, D., 1996, *Geographic Information Systems: Socioeconomic Applications* (London: Routledge).

Marx, R. W., 1986, The TIGER system: automating the geographic structure of the United States census. Reprinted in *Introductory Readings in Geographic Information Systems*, 1990, edited by D. J. Peuquet and D. F. Marble (London: Taylor and Francis), pp. 120–141.

Morehouse, S., 1992, The ARC/INFO geographic information system. *Computers & Geosciences*, **18**, 435–441.

Okabe, A., and Miller, H. J., 1996, Exact computational methods for calculating distance between objects in a cartographic database. *Cartography and Geographic Information Systems*, **23**, 180–195.

Peucker, T. K., and Chrisman, N., 1975, Cartographic data structures. *The American Cartographer*, **2**, 55–69.

Peuquet, D. J., 1984, A conceptual framework and comparison of spatial data models. *Cartographica*, **21**, 66–113.

Peuquet, D. J., and Zhan, C. X., 1987, An algorithm to determine the directional relationship between arbitrarily-shaped polygons in the plane. *Pattern Recognition*, **20**, 65–74.

Raper, J. F., and Maguire, D. J., 1992, Design models and functionality in GIS. *Computers & Geosciences*, **18**, 387–394.

Reed, C., 1999, GIS Users shouldn't forget about topology. *GeoWorld*, **April**, 12.

Strand, E., 1998, Shapefiles shape GIS data transfer standards. *GIS World*, **May**, 28–29.

Theobald, D. M., and Hobbs, N. T., 2001, Functional definition of landscape structure using a gradient-based approach. In *Predicting Plant and Animal Occurrences: Issues of Scale and Accuracy*, edited by J. M. Scott, P. J. Heglund, M. Morrison, M. Raphael, J. Haufler and B. Wall (Covello, CA: Island Press).

Tomlinson, R. 1967, *An Introduction to the Geo-Information System of the Canada Land Inventory* (Ottawa: Canada Department of Forestry and Rural Development).

Tomlinson, R., 1998, The Canada Geographic Information System. *The History of Geographic Information Systems: Perspectives from the Pioneer*, edited by T. W. Foresman (Upper Saddle River, NJ: Prentice Hall), pp. 21–32.

White, D., Corson-Rikert, J., and Maizel, M., 1987, 'WYSIWYG' map digitizing: Real time geometric correction and topological encoding. In *Proceedings from Auto-Carto 8* (Bethesda, MD: American Congress on Surveying and Mapping), pp. 739–743.

Wiens, J. A., Stenseth, N. C., van Horne, B., and Ohms, R. A., 1993, Ecological mechanisms and landscape ecology. *Oikos*, **66**, 369–380.

Worboys, M. F., 1995, *GIS: A computing perspective* (London: Taylor & Francis).

Appendix. Algorithms to compute proximal connectivity

Also, see: http://www.ndis.nrel.colostate.edu/davet/topology/connectivity.html for a listing of Avenue code.

Algorithm 1: Brute-force

```
D = distance threshold, D > = 0
F = { features } ' F is a list of all features
C = {}      ' C is an empty list that will contain connected pairs of features
for every f1 in F
 for every f2 in F
  if ( f1 is within distance D of f2 ) then
   add f1,f2 to C
  End if
 End for f2
End For f1
```

Algorithm 2: Brute-force reflexive

```
D = distance threshold, D > = 0
F = { features } ' F is a list of all features
C = {}      ' C is an empty list that will contain connected pairs of features
for every f1 in F
 for every f2 in f1..F   ' only step through unvisited features in F
  if ( f1 is within distance D of f2 ) then
   add f1,f2 to C
  End if
 End for f2
End For f1
```

Algorithm 3: Recursive subdivision

```
D = distance threshold, D > = 0
F = { features } ' F is a list of all features
C = {}      ' C is an empty list that will contain connected pairs of features
x = threshold of number of features, 10 < = x < = 100 or so
S = { map extent }' a last-in, first out stack of map extent
```

```
while F is not empty
 X = F within S        ' select features within map extent in S
   if ( x < number of X ) then
     ' divide extent S into 4 tiles s1, s2, s3, s4
     push s1, s2, s3, and s4 onto S
  else    ' find relations by brute-force
     for every f1 in x
       for every f2 in f1..x   ' only step through unvisited features in F
         if ( f1 is within distance D of f2 ) then
           add f1,f2 to C
     End if
       End for f2
     End For f1
   End if
End while
```